REMARKS

Applicant respectfully requests reconsideration and allowance of the subject application. Claims 13-24 are pending, of which claims 22-24 have been amended. The amendment to claims 22-24 are simply to correct an informality, and not to overcome prior art.

35 U.S.C. §112 Claim Rejections

Claims 22-24 are rejected under 35 U.S.C. §112, second paragraph, as being indefinite (*Office Action* p.3). Appropriate amendments to claims 22-24 have been provided herein and Applicant respectfully requests that the §112 rejection be withdrawn.

35 U.S.C. §102 Claim Rejections

Claims 13-24 are rejected under 35 U.S.C. §102(b) as being anticipated by Hendren et al., "Supporting Array Dependence Testing for an Optimizing/ Parallelizing *C* Compiler" (1993) (hereinafter, "Hendren") (*Office Action* p.4). Applicant respectfully traverses the rejection.

As the title indicates, Hendren describes features for a *C* compiler, such as support analyses in the context of *C* parallelizing compilers to handle loops and aliasing due to pointers (*Hendren* p.2, ¶1). Specifically, Hendren describes a framework designed to handle the complexities of the *C* language that affect array dependence analysis (*Hendren* p.3, ¶1.2). Hendren illustrates in Fig. 2 and describes that *C* program files are input to a compiler which produces a simplified

and compositional structured representation called "SIMPLE" to handle various complications in the *C* program files (*Hendren* p.4, ¶2; Fig. 2).

In the Background section of the present application, Applicant describes compiling a high-level programming language into machine instructions where the compiling includes lexical, syntax, and semantic analysis, as well as generating a syntax tree during compiling (*Specification* p.1, line 19 to p.2, line 18). In contrast to a compiler that translates a program written in a *specific* programming language (e.g., the *C* programming language) into another programming language or machine code, as described in Hendren, Applicant describes that a program can be developed and represented by a "high-level program tree that is a *syntax-independent* representation" of a programmer's intent (*Specification* p.4, lines 33-37). A programmer can directly manipulate the syntax-independent program tree, "which is in contrast to conventional programming systems in which a programmer manipulates a textual representation of the program that is later converted into a syntax tree during compilation" – as described in Hendren. (*Specification* p.5, lines 1-5).

Claim 13 recites a method comprising "identifying a syntax-independent programming intent represented as a first node of a data structure". Hendren does not show or disclose a syntax-independent programming intent, or identifying a syntax-independent programming intent represented as a node of a data structure, as recited in claim 13.

As described above, Hendren is syntax-dependent and describes an intermediate structure to compile *C* program files (*Hendren* p.4, ¶2; Fig. 2). The

Office cites Hendren for teaching a syntax-independent programming intent represented as a first node of a data structure (*Office Action* p.4). Applicant disagrees because what Hendren illustrates in Fig. 7(b) is a general expression tree that represents an index in an example simplified program illustrated in Fig. 7(a) (*Hendren* pp.9-10, ¶5.1; Fig. 7). Hendren describes that a general expression tree is built during the structured "SIMPLE" analysis when compiling the *C* program files (*Hendren* p.4, ¶2; Fig. 2; p.9, ¶5.1).

As described in Hendren, the general expression tree is generated from a source program during compiling. To the contrary, Applicant claims identifying a syntax-independent programming intent represented as a data structure node. Accordingly, claim 13 along with dependent claims 14-16 are allowable over Hendren and Applicant respectfully requests that the §102 rejection be withdrawn.

Claim 17 recites a method of handling data comprising "reading a syntax-independent programming intent represented as a first node of a hierarchical tree". Hendren does not show or disclose a syntax-independent programming intent, or reading a syntax-independent programming intent represented as a node of a hierarchical tree, as recited in claim 17.

As described above in the response to the rejection of claim 13, Hendren is syntax-dependent and describes an intermediate structure to compile *C* program files (*Hendren* p.4, ¶2; Fig. 2). Further, Hendren describes that a compiler generates a general expression tree from a source program during compiling (*Hendren* p.4, ¶2; Fig. 2; pp.9-10, ¶5.1). Hendren also does not disclose a syntax-independent programming intent, as recited in claim 17.

Further, the Office rejects claim 17 as having corresponding functionality to claim 13 (*Office Action* p.7). Applicant disagrees that "reading a syntax-independent programming intent", as recited in claim 17, is a corresponding function to "identifying a syntax-independent programming intent", as recited in claim 13. The Office has not provided a basis for the rejection of "*reading* a syntax-independent programming intent" other than to state vaguely that this is "corresponding functionality" (*Office Action* p.5).

Accordingly, claim 17 along with dependent claims 18-20 are allowable over Hendren and Applicant respectfully requests that the §102 rejection be withdrawn.

Claim 21 recites a data structure comprising "a first node received as an input and configured for display as a representation of a syntax-independent programming intent". Hendren does not show or disclose a node of a data structure "received as an input and configured for display as a representation of a syntax-independent programming intent", as recited in claim 21.

As described above in the response to the rejection of claim 13, Hendren is syntax-dependent and describes an intermediate structure to compile *C* program files (*Hendren* p.4, ¶2; Fig. 2). Hendren describes that a compiler generates a general expression tree from a source program during compiling (*Hendren* p.4, ¶2; Fig. 2; pp.9-10, ¶5.1).

Further, Hendren does not show or disclose that a node of a data structure can be received as an input, or that a node received as an input is configured for display. Although Hendren illustrates the general expression tree in Fig. 7(b),

there is no indication whatsoever that the general expression tree is configured for display. Accordingly, claim 21 along with dependent claims 22-24 are allowable over Hendren and Applicant respectfully requests that the §102 rejection be withdrawn.

Conclusion

Pending claims 13-24 are in condition for allowance. Applicant respectfully requests reconsideration and issuance of the subject application. If any issues remain that preclude issuance of this application, the Examiner is urged to contact the undersigned attorney before issuing a subsequent Action.

Respectfully Submitted,

Dated: Jun 15, 2004

By: /David Morasch/

David A. Morasch
Reg. No. 42,905
(509) 324-9256 x 210